SE1011 Exam 2 Feedback

Dr. Yoder, Fall 2014, MSOE

Problem 1

① -0 See the solution shown in class for a simpler solution to this problem. Only one loop is needed. You don't need to implement the 0 and 1 cases separately – you just need to mentally run the program to check that it works for those cases.

② -0.5 Multiplies by x one too many times. For y == 0, the result is 1 and for y == 1, the result is x.

③ -1 Does not specify initial value for result variable. The first time `result *= x;` is used, result will not have had a value assigned, leading to a compile-time error.

④ Use == for equality comparison of primitive types, not =.

⑤ -2 Counts to y correctly, but instead of multiplying by x with each iteration, does something else. The body of the loop should read something like `result *= x;` or `result = result * x;`

⑥ -1 Loop does not count correctly, but appears to attempt to count to y.

⑦ -2 Loop counts to something unrelated to the problem.

⑧ -3 Body of loop does not multiply by x at all.

Problem 2

① -1 Use == for reference comparison (e.g. `str == null`) and .equals for object comparison (e.g. `str.equals("")`)

② -0 it is clearer to put the final dialog after the loop, to show that it will always happen.

③ -0 it is clearer to move the condition of the if to be the condition of the loop, to make it clearer when the loop exits.

④ -0.5 Use == for comparison, = for assignment.

⑤ -1 JOptionPane.showInputDialog returns a String – you cannot use the console scanner to access what the user types in the GUI dialog.

⑥ -0.5 Must check for null first to avoid null-pointer exception.

Problem 5b

① You can simplify code like

```
boolean areEqual;
if(this.velocityMps == other.velocityMps) {
   areEqual = true;
} else {
  areEqual = false;
}
```

```
return true;
```

to simply be

```
return this.velocityMps == other.velocityMps;
```

because `this.velocityMps == other.velocityMps` is a true-false (boolean) expression.